ELSEVIER

European Journal of Operational Research 126 (2000) 80–88

EUROPEAN JOURNAL OF OPERATIONAL RESEARCH

www.elsevier.com/locate/dsw

Theory and Methodology

Allocation of tasks to specialized processors: A planning approach

K.J. Becker [a], D.P. Gaver [b], K.D. Glazebrook [c], P.A. Jacobs [b], S. Lawphongpanich [b,*]

[a] *Compaq Computer Corporation, Houston, TX 77070, USA*
[b] *Department of Operations Research, Naval Postgraduate School, 1411 Cunningham Road, Monterey, CA 93943, USA*
[c] *Department of Statistics, University of Newcastle upon Tyne, Newcastle upon Tyne NE1 7RU, UK*

Received 1 November 1998; accepted 1 April 1999

Abstract

This paper addresses the problem of scheduling randomly arriving tasks of different types at a diversified service system. Servers at such a system differ in that each may specialize in one task type, but can also perform others perhaps less rapidly and adequately than does a specialist. We consider the issue of how much redirection of tasks from specialists to non-specialists may be desirable in such a system and propose a static model in which tasks are randomly assigned to servers. Two scheduling strategies for individual servers are also considered: one in which each server performs the tasks assigned to him or her in order of their arrival and the second in which each server schedules his or her workload optimally. The problems for finding the best random assignment probabilities are formulated as mathematical programs. Results from a numerical example provide information that is both informative and useful in decision-making.

Keywords: Queuing; Job allocation; Service System; Optimization

1. Introduction

Consider a diversified service system whose constituent service facilities or servers are unequal in their capacities (or training) to perform different tasks. Usually, a type-j task is most expeditiously handled by server $i(j)$, and quantifiably less so by others. Although there are many examples (see, e.g., Stanford and Grassmann, 1993; Green, 1985) for such a system, our work was motivated by call centers (Mehrotra, 1997) of companies producing multiple types of products. Customers telephone these centers for service or technical support. These telephone calls are then routed or assigned to agents or technicians. Preferably, calls concerning, say, product A should be handled or routed to agents or technicians with expertise in product A. This practice ensures that calls are handled properly and efficiently. When demands for support are not uniform across the company's products, such a policy may overload the servers with expertise in, say, new products with a higher sales volume. To lessen the load on these servers

* Corresponding author.
E-mail address: slawphon@nps.navy.mil (S. Lawphongpanich).

0377-2217/00/$ - see front matter
PII: S0377-2217(99)00282-9

and, perhaps, to shorten the customer waiting time, some calls may be assigned or routed to less efficient or non-expert servers.

There are both static and dynamic strategies for assigning tasks (or calls) to or balancing workloads among servers. Dynamic strategies generally offer the possibility of improved task assignment, and possibly reassignment, at the expense of additional information, communication, and processing overhead. For call centers, this overhead may not be too expensive. A number of computer telephony integration (CTI) systems monitor and frequently provide the status for servers and call center operations. However, developing a strategy that optimally takes advantage of information provided by these systems is involved. Many commercial CTI systems seem to offer automatic call distribution or routing without revealing any systematic methodology for determining an 'optimal' strategy. On the other hand, simple myopic distribution or routing policies based on the latest system status may be self-defeating under heavy call traffic.

In the literature, there are several papers on dynamic load balancing strategies for similar systems. Gaver et al. (1993) consider dynamic complete-service assignment rules for a generalized repairman problem. For distributed computer systems, Eager et al. (1986a,b) and Zhou (1988) propose load-balancing heuristics that reduce response times in moderate to heavy traffic. Perhaps, ideas in these papers can be applied to call centers; they are worthy topics for future investigation.

This paper focuses on static strategies. Generally, they allocate tasks to servers in a deterministic or randomized fashion, typically independent of the system status. As such, these strategies are reasonable for situations in which the system status is not readily available or too expensive and difficult to monitor and control well in real time.

For call centers, some commercial CTI systems allow distribution or routing parameters to be adjusted or modified periodically, perhaps in response to the newly updated system status or forecasted system parameters, e.g., call arrival rates. In such systems, static strategies can be used to determine the 'optimal' (routing) parameter adjustments periodically. Implemented in this manner, a frequently recomputed static strategy exhibits a dynamic, but not totally myopic, behavior. Moreover, static strategies are useful for skill-based workforce management, scheduling, and in setting service objectives. These strategies also lead to technically manageable models for quantifying the benefits of additional server training to improve service times or to acquire new skills.

One static strategy is to assign an incoming type-j task to server i with probability a_{ji}. Intuitively, a_{ji} is simply a fraction or percentage of type-j tasks that is allocated to server i. (If $a_{ji} = 1$ for $i = i(j)$ and $a_{ji} = 0$ for $i \neq i(j)$, for every j, then the assignment is deterministic.) This approach allows the service stations to be treated as a set of independent M/G/1 queues. To determine the proper values for the decision variables a_{ji}, this paper relies on a combination of results from classical M/G/1 queuing theory and from optimization. The use of optimization techniques to allocate resources in queues is not new. For example, Shanthikumar and Yao (1992) describe a mathematical programming approach to scheduling control in the context of multi-class queuing systems; Ross and Yao (1991) study the problem of balancing workloads on computers connected by a network; and Berman et al. (1990) address optimal location problems in a stochastic environment. In these studies, the resulting optimization problems are often difficult and require specialized or heuristic techniques to obtain solutions, optimal or otherwise. The optimization problems in this paper are relatively simple and can be effectively and efficiently solved using commonly available software.

In Section 2, we present the model, provide basic results, and pose a generic optimization problem whose solution will yield the assignment probabilities. Using queuing theory, Section 3 develops two delay-based objective functions for the optimization problem. A numerical study is described in Section 4 and the paper concludes with a discussion in Section 5 of alternative performance measures that penalize long delays more severely.

2. Model

Assume that there are J task types and I servers, each with different expertise. Tasks (calls) arrive at the system as a Poisson process with rate λ. An arriving task is of type j, where $j \in \{1,\ldots,J\}$, with an independent probability p_j. The ability of each server to perform tasks of different types is reflected in his or her service time for each task type. The service time for a type-j task by server i is denoted by S_{ji} and may vary systematically with i for a variety of reasons, one being the extent of the ith server's training for or experience with type-j tasks. When server i is unable to handle type-j tasks, S_{ji} is assumed to be infinite almost surely. This is a case of total incompatibility and certainly exists in many practical settings.

When the decision variables a_{ji} are given, the tasks arrive at server i, $i \in \{1,\ldots,I\}$, at rate λ_i, where

$$\lambda_i = \lambda \sum_{j=1}^{J} p_j a_{ji} \equiv \bar{p}_i. \tag{2.1}$$

By standard results, λ_i is the rate of a process independent of those prevailing for other servers ($\neq i$). In addition, moments of the so-called effective service time, S_i, of tasks arriving at server i can be computed. An incoming task is of type j with probability p_j and it experiences service time S_{ji}, if it is dispatched to server i. Hence,

$$\text{Prob}(S_i = S_{ji}) = \frac{p_j a_{ji}}{\bar{p}_i}.$$

The first two moments of S_i can be calculated by conditioning, i.e.,

$$E[S_i] = \sum_{j=1}^{J} E\left[S_{ji}\right] p_j a_{ji} \Big/ \bar{p}_i, \tag{2.2}$$

$$E\left[S_i^2\right] = \sum_{j=1}^{J} E\left[S_{ji}^2\right] p_j a_{ji} \Big/ \bar{p}_i. \tag{2.3}$$

From Eq. (2.1) and (2.2), the ith server's traffic intensity is given by

$$\rho_i = \lambda_i E[S_i] = \lambda \bar{p}_i E[S_i] = \lambda \sum_{j=1}^{J} E\left[S_{ji}\right] p_j a_{ji}. \tag{2.4}$$

To be a valid set of assignment probabilities, each a_{ji} must be nonnegative such that $\sum_{i=1}^{I} a_{ji} = 1$, for all j. To ensure that each server can complete his or her assigned tasks, the resulting ρ_i in Eq. (2.4) must be less than 1 for all i. This last condition renders all deterministic assignments infeasible in certain cases. In particular, if $\lambda p_j E[S_{ji}] \geqslant 1$ for all i, then no one server can handle all type-j tasks and they must be distributed among several servers.

In order to determine a set of assignment probabilities, we formulate an optimization problem of the following kind:

$$\begin{aligned} \text{P}: \quad & \min_{a_{ji}} f(\underline{a}) \\ \text{s.t.} \quad & \\ & \lambda \sum_{j=1}^{J} p_j a_{ji} E\left[S_{ji}\right] < 1, \quad i = 1,\ldots,I, \\ & \sum_{i=1}^{I} a_{ji} = 1, \qquad j = 1,\ldots,J, \\ & a_{ji} \geqslant 0, \quad i = 1,\ldots,I;\ j = 1,\ldots,J, \end{aligned} \tag{2.5}$$

where $f(\underline{a})$ denotes a measure of system performance. Although technically correct, Eq. (2.5) is not numerically implementable and is usually replaced by an inequality of the form

$$\lambda \sum_{j=1}^{J} p_j a_{ji} E\left[S_{ji}\right] \leqslant 1 - \varepsilon, \quad i = 1,\ldots,I, \tag{2.5a}$$

for some small $\varepsilon > 0$.

3. Total task delay

One standard choice of performance measure $f(\underline{a})$ for problem P is a total of expected delays. Clearly, delays depend upon the a_{ji}. However, the scheduling strategy used by each server will also have an effect. Below, two scheduling strategies are examined. One makes the standard assumption that each server performs his or her assigned tasks

according to their arrival order, i.e., first-come-first-serve (FCFS), while the other allows each server to schedule the assigned tasks to minimize his or her own expected weighted delay.

Many call centers adopt FCFS, for it is believed to be a fair way to treat customers. To obtain the total expected delay under FCFS, the Pollaczek–Khintchine–Kendall formula provides the expected long-run waiting time $E[W]$ for an M/G/1 system. In our case, the expected long-run waiting time for server i is

$$E[W_i] = \lambda_i E\left[S_i^2\right] \frac{1}{2(1-\rho_i)}, \quad \text{if } \rho_i < 1, \tag{3.1}$$

where $E[S_i^2]$ and ρ_i are as given in Eq. (2.3) and (2.4), respectively. The total expected delay for a type-j task is easily seen to be

$$E\left[D_j\right] = \sum_{i=1}^{I} a_{ji}\left(E[W_i] + E\left[S_{ji}\right]\right). \tag{3.2}$$

Note that Eq. (3.2) becomes infinite if any $\rho \geqslant 1$. The total expected long-run *weighted* delay (per unit time) under FCFS is

$$\begin{aligned} f(\underline{a}) &\equiv D^{\mathrm{FCFS}}(\underline{a}) \\ &= \sum_{j=1}^{J} \lambda p_j b_j E[D_j] \\ &= \sum_{j=1}^{J} \lambda p_j b_j \left[\sum_{i=1}^{I} a_{ji}\left(E[W_i] + E\left[S_{ji}\right]\right)\right], \end{aligned}$$

where b_j is the weight for the delay of type-j tasks.

Under FCFS, customers may experience longer waiting times than necessary. Following the approach in Ross and Yao (1991), a natural extension of the above model is to allow each server to schedule his or her assigned tasks optimally. Then, the associated performance measure can be expressed as

$$f(\underline{a}) \equiv D^{\mathrm{OPT}}(\underline{a}) = \sum_{i=1}^{I} D_i^{\mathrm{OPT}}(\underline{a}_i),$$

where $D_i^{\mathrm{OPT}}(\underline{a}_i)$ is the total expected weighted delay arising from server i when the latter schedules his or her traffic optimally. The optimization concerned is over the class of work-conserving strategies with priorities imposed in a non-preemptive fashion (i.e., a task, once started, is always processed to completion before any other task is granted access to the server). A classic result due to Fife (1965) (see also Kleinrock, 1976) indicates that the optimal scheduling strategy for server i takes the following simple form: renumber the task types such that

$$\frac{b_1}{E[S_{1i}]} \leqslant \frac{b_2}{E[S_{2i}]} \leqslant \cdots \leqslant \frac{b_J}{E[S_{Ji}]}.$$

It is optimal for server i to process tasks in decreasing numerical order of the type identifier. Hence at each decision epoch, server i will next choose to serve the task with the largest type number present in the system. It is a consequence of the ground breaking analysis of this result based on work conservation principles in Gelenbe and Mitrani (1980) that $D_i^{\mathrm{OPT}}(\underline{a}_i)$ is given by the following expression:

$$\begin{aligned} D_i^{\mathrm{OPT}}(\underline{a}_i) = {} & \frac{b_1}{E[S_{1i}]} g(\{1,2,\ldots,J\}) \\ & + \left[\frac{b_2}{E[S_{2i}]} - \frac{b_1}{E[S_{1i}]}\right] g(\{2,3,\ldots,J\}) \\ & + \left[\frac{b_3}{E[S_{3i}]} - \frac{b_2}{E[S_{2i}]}\right] g(\{3,4,\ldots,J\}) \\ & + \cdots + \left[\frac{b_J}{E[S_{Ji}]} - \frac{b_{(J-1)}}{E[S_{(J-1)i}]}\right] g(\{J\}), \end{aligned}$$

where

$$\begin{aligned} g(\Omega) = {} & \frac{\left(\frac{1}{2}\sum_{j=1}^{J} \lambda p_j a_{ji} E[S_{ji}^2]\right)\left(\sum_{j\in\Omega} \lambda p_j a_{ji} E[S_{ji}]\right)}{\left(1 - \sum_{j\in\Omega} \lambda p_j a_{ji} E[S_{ji}]\right)} \\ & + \sum_{j\in\Omega} \lambda p_j a_{ji} (E[S_{ji}])^2 \end{aligned}$$

for any $\Omega \subseteq \{1,2,\ldots,J\}$.

Clearly, $D^{\mathrm{OPT}}(\underline{a}) \leqslant D^{\mathrm{FCFS}}(\underline{a})$ for all $(\underline{a})$, since FCFS is a feasible task schedule. However, it is not clear that call centers should switch to the logistically more complex optimal scheduling strategy based on this result alone. The next section contains an illustrative numerical comparison between the two approaches to local scheduling.

4. Numerical study

Assume that the service time, S_{ji}, has a gamma distribution with parameters α_i and $k_{ij}\beta_i$. Under this assumption, the first two moments can be written as

$$E[S_{ij}] = k_{ij}\beta_i/\alpha_i$$

and

$$E[S_{ji}^2] = \left(\left(k_{ji}\beta_i\right)^2 + k_{ji}\beta_i\right)/\alpha_i^2.$$

For our example, $k_{ji} = |j-i|+1$. Under this choice of k_{ji}, every server is capable of performing all J tasks, but not necessarily at the same level of efficiency. The mean and variance of a service time for type-j tasks by server i are the smallest when $j=i$. In this sense, we consider server i as the expert for type-i tasks. For others, server i takes longer to perform a type-j task when $j \neq i$ and the expected service time depends on the difference between j and i. A larger difference means a longer expected service time.

Problem P in Section 2 and the total task delay functions for the two scheduling strategies in Section 3 (i.e., the optimal strategy and FCFS) only make use of the first two moments of the service time distribution. Hence, the optimal assignment probabilities depend only upon the first two moments. So, as long as its first two moments match the desired values, the gamma distribution is adequate for our study and provides considerable flexibility in modeling service times. However, any other distributional assumption yielding the same first and second moments would also generate identical results throughout. Of course, the same generally does not hold for other objective functions or performance measures, e.g., those described in the next section.

Although somewhat arbitrary, our choice for k_{ij}'s is intended mainly to distinguish the skills or capabilities among the servers. Other choices for k_{ij}'s are possible and would have a significant impact on the optimal assignment probabilities. Different values for k_{ij}'s would yield different first and second moments for the service times. In turn, these moments would generate different task delay functions and optimal assignment probabilities. For our study, $I=5$, $J=5$, $\alpha_i=2$, $\beta_i=1$, and $b_j=1$. (Recall that b_j's are weights for task delays.) Also, the arriving tasks are of type j with probability $p_j = 0.35 - 0.05j$ for $j=1,2,\ldots,5$. Thus, type-1 tasks arrive at the system or call center most frequently and type-5 tasks, the least frequently.

To select meaningful arrival rates for our study, we first determine the maximum arrival rate that allows every server's traffic intensity to be no greater than one. This is accomplished by solving the following optimization problem.

$$\lambda_{\max} = \max_{\lambda, a_{ji}} \lambda$$

s.t.

$$\lambda \sum_{j=1}^{J} p_j a_{ji} E\left[S_{ji}\right] \leqslant 1, \quad i = 1,\ldots,I,$$

$$\sum_{i=1}^{I} a_{ji} = 1, \quad j = 1,\ldots,J,$$

$$a_{ji} \geqslant 0, \quad i = 1,\ldots,I;\ j = 1,\ldots,J.$$

Intuitively, $\lambda_{\max}$ is the maximum task traffic that can be admitted or is feasible to the system.

Initially, the arrival rate λ is set at 7.88 or 95% of $\lambda_{\max}$ ($=8.29$). Problem P, with $\varepsilon = 0.01$ in Eq. (2.5a), was solved twice using a commercial optimization package called GAMS (see Brooke et al., 1992), once with $f(\underline{a}) = D^{\mathrm{FCFS}}(\underline{a})$ and the other with $f(\underline{a}) = D^{\mathrm{OPT}}(\underline{a})$. In general, neither $D^{\mathrm{FCFS}}(\underline{a})$ nor $D^{\mathrm{OPT}}(\underline{a})$ is convex and a solution found by GAMS may not be globally optimal. In fact, GAMS can only guarantee locally optimal solutions to problem P when the objective function $f(\underline{a})$ is not convex. In an attempt to find a good solution for each objective, 10 sets of randomly generated assignment probabilities were used to start or initialize the optimization algorithm. Each set of initial assignment probabilities may or may not generate a new locally optimal solution to problem P. The assignment probabilities displayed in Tables 1 and 2 are the locally optimal solutions with the smallest $D^{\mathrm{FCFS}}(\underline{a})$ and $D^{\mathrm{OPT}}(\underline{a})$, respectively.

Table 1
Optimal task assignments using FCFS strategy

	Server 1	Server 2	Server 3	Server 4	Server 5
Type-1 task	0.8158		0.0256	0.0705	0.0881
Type-2 task		0.9705	0.0295		
Type-3 task			1.0000		
Type-4 task				1.0000	
Type-5 task					1.0000

Table 2
Optimal task assignments using optimal scheduling strategy

	Server 1	Server 2	Server 3	Server 4	Server 5
Type-1 task	0.8085	0.0082	0.0176	0.0744	0.0913
Type-2 task		0.9488	0.0512		
Type-3 task			1.0000		
Type-4 task				1.0000	
Type-5 task					1.0000

Table 3
Percentage of tasks assigned to non-expert servers

δ	λ_{max}	λ used	Percent of λ_{max}	FCFS strategy		Optimal scheduling strategy	
				Expected delay	Percent of tasks to non-experts	Expected delay	Percent of tasks to non-experts
0.2	9.30	8.84	95	92.14	8.32	81.68	9.13
0.4	8.91	8.46	95	90.84	6.84	75.44	7.42
0.6	8.64	8.21	95	90.18	5.78	71.92	6.30
0.8	8.44	8.02	95	89.74	4.94	68.86	5.46
1.0	8.29	7.88	95	89.47	4.27	66.67	4.85
1.2	8.17	7.76	95	89.28	3.72	65.29	4.34
1.4	8.07	7.67	95	89.16	3.26	64.40	3.88
1.6	7.99	7.59	95	89.72	2.99	63.65	3.49
1.8	7.91	7.52	95	89.57	2.81	61.94	3.14
2.0	7.84	7.45	95	89.40	2.64	60.42	2.88

For the assignment probabilities in Tables 1 and 2, the total expected delay is 89.47 under FCFS and 66.67 under the optimal scheduling strategy. (See Tables 3 and 4 below.) Although server 1 is the expert at performing type-1 tasks, he or she cannot service all of them because $\lambda p_1 E[S_{11}] > 1$. Hence, some type-1 tasks must be assigned to non-expert servers. If all of type-2 tasks are allocated to server 2, his or her traffic intensity is 0.99. In order to minimize total expected delay, it is also advantageous to allocate some type-2 tasks to non-expert servers.

The diagonal elements in Table 2 are slightly smaller that those of Table 1. This indicates that the quantity of work allocated to non-expert servers is slightly greater when servers schedule their own assigned tasks optimally. Averaging over the diagonal elements of Table 1, the FCFS strategy allocates 95.73% of the tasks to the experts and the remaining 4.27% to non-expert

Table 4
Total expected delay with varying arrival rates

λ	Percent of $\lambda_{\max}$	$D^{\mathrm{FCFS}}(\underline{a}^{\mathrm{FCFS}})$	$D^{\mathrm{OPT}}(\underline{a}^{\mathrm{FCFS}})$	$D^{\mathrm{OPT}}(\underline{a}^{\mathrm{OPT}})$	$100\frac{[D^{\mathrm{FCFS}}(\underline{a}^{\mathrm{FCFS}})-D^{\mathrm{OPT}}(\underline{a}^{\mathrm{OPT}})]}{D^{\mathrm{FCFS}}(\underline{a}^{\mathrm{FCFS}})}$
4.15	50	4.14	4.14	4.14	0.00
4.56	55	5.15	5.15	5.14	0.07
4.98	60	6.45	6.43	6.42	0.63
5.39	65	8.14	8.03	8.00	1.78
5.81	70	10.40	10.11	10.03	3.54
6.22	75	13.56	12.92	12.74	6.00
6.63	80	18.30	16.96	16.61	9.22
7.05	85	26.20	23.39	22.72	13.31
7.46	90	42.02	35.61	34.22	18.56
7.88	95	89.47	70.48	66.67	25.48
8.21	99	523.99	326.17	326.17	37.75

servers. The corresponding figures for the optimal scheduling strategy are 95.15% and 4.85%, respectively.

When $|j-i| = |j-n|$, our choice of k_{ji} implies that $E[S_{ji}] = E[S_{jn}]$ and $E[S_{ji}^2] = E[S_{jn}^2]$, i.e., server i and n are equally capable at performing type-j tasks. However, it is interesting to note that there is a distinct server preference for the assignments in Tables 1 and 2, in that $a_{ji} = 0$ if $j > i$. To explain this preference, consider servers 1 and 3 as non-expert servers for type-2 tasks. With respect to these tasks, servers 1 and 3 are equally capable. However, the effective arrival rate for type-3 tasks, for which server 3 is the expert, is less than that of type-1 tasks, for which server 1 is the expert. Since $E[S_{ii}]$ and $E[S_{ii}^2]$ are 0.5 and 1.5 for all i, server 3 is less busy with type-3 tasks than server 1 with type-1 tasks. Thus, there would be less delay by sending type-2 tasks to server 3 instead of server 1.

The quantity of work allocated to non-expert servers also depends on their ability to service the tasks for which they are not skilled or not properly trained. In our numerical example, this lack of ability grows with the parameter k_{ji}. To examine its effects on the task assignment, k_{ji} is now given a more general form $\delta|j-i|+1$. Table 3 displays the percentage of work assigned to non-expert servers for various values of δ. As before, each solution shown in the table is the best local optimal solution among those obtained by solving problem P using ten randomly chosen initial solutions.

As δ increases, server i is less versatile and takes more time to service a type-j task, where $j \neq i$. So, it becomes less beneficial to assign tasks to non-expert servers. For FCFS, the percentage of tasks assigned to non-expert servers decreases from 8.32% to 2.64% as δ increases from 0.2 to 2.0. We observe a similar phenomenon under the optimal scheduling strategy.

In Table 3, the task arrival rate is always 95% of $\lambda_{\max}$. As δ increases, tasks take more time to complete at non-expert servers and it becomes less beneficial to redirect tasks to them. So, the maximum task traffic that can be admitted or is feasible to the system is restricted mainly by the amount of work that the experts can handle. Therefore, $\lambda_{\max}$ decreases as δ increases. Since the λ in problem P is always 95% of $\lambda_{\max}$, a smaller $\lambda_{\max}$ means that a smaller number of tasks arrive at the system. Next, observe that our choices for the first two moments of the service time are independent of $\lambda_{\max}$. As δ increases (i.e., $\lambda_{\max}$ decreases), the expert servers, servers 1 and 2 in particular, can complete higher percentage of their tasks and redirect less of them to other (non-expert) servers. So, tasks are completed more quickly and the total expected delay decreases as δ increases.

Let $(\underline{a})^{\mathrm{FCFS}}$ and $(\underline{a})^{\mathrm{OPT}}$ denote the 'best' assignment probabilities under FCFS and the optimal scheduling strategy, respectively. As before, the 'best assignment' refers to the best solution among those obtained by solving problem P using ten randomly generated initial assignment proba-

bilities. Table 4 displays the values of $D^{\mathrm{FCFS}}(\underline{a}^{\mathrm{FCFS}})$, $D^{\mathrm{OPT}}(\underline{a}^{\mathrm{FCFS}})$, and $D^{\mathrm{OPT}}(\underline{a}^{\mathrm{OPT}})$ for different arrival rates and δ set equal to 1. Note that $D^{\mathrm{OPT}}(\underline{a}^{\mathrm{FCFS}})$ is the total expected delay using the FCFS assignment probabilities while allowing servers to schedule their assigned tasks optimally.

As listed in the 3rd and 5th columns of Table 4, the optimal total expected delays using the two scheduling strategies are similar when the arrival rate is small. In particular, all three values, i.e., of $D^{\mathrm{FCFS}}(\underline{a}^{\mathrm{FCFS}})$, $D^{\mathrm{OPT}}(\underline{a}^{\mathrm{FCFS}})$, and $D^{\mathrm{OPT}}(\underline{a}^{\mathrm{OPT}})$, are the same when $\lambda = 4.14$ or 50% of $\lambda_{\max}$. On the other hand, as shown in the 6th column of Table 4, the relative difference or efficiency gained by using the optimal scheduling strategy increases as the arrival rate increases. In other words, the scheduling strategy has little effect on the delay or optimal task allocation in light traffic. As the traffic becomes heavier, the total expected delay improves significantly by using the optimal scheduling strategy. Naturally, in the heavy traffic limit ($\rho_i = 1$), all three delays would eventually become infinite. When $\lambda = 8.21$ or 99% of $\lambda_{\max}$, $\underline{a}^{\mathrm{FCFS}}$ equals $\underline{a}^{\mathrm{OPT}}$ in our example.

It is also interesting to compare $D^{\mathrm{OPT}}(\underline{a}^{\mathrm{FCFS}})$ and $D^{\mathrm{OPT}}(\underline{a}^{\mathrm{OPT}})$, the 4th and 5th columns of Table 4. Empirically, the results suggest that the FCFS assignment is relatively robust for our example, in the sense that it yields similar total expected delays when the servers schedule optimally. Our calculations suggest that the FCFS assignment performs particularly well in very light and very heavy traffic. When the traffic is moderate to heavy, then the optimal scheduling strategy yields less delay in the example. However, it should be noted that our model does not reflect the effort or resources needed to implement the optimal scheduling strategy by each server.

5. Extensions and conclusion

There are a number of performance measures that can be used as an objective function for problem P. In some cases, it is important that the penalty for task delays be more stringent for long tasks, and in a manner that the (linear) long-run expectation of total delay does not reflect adequately. One such penalty parameterization is exponential with the analysis focusing on $E[\mathrm{e}^{\theta_i D_i}]$, with positive θ_i, rather than on $E[D_i]$. Classical M/G/1 theory says that the limiting transform of W_i is given by

$$E[e^{\theta_i W_i}] = (1-\rho_i) \Big/ \left(1 - \frac{\rho_i\{E[\mathrm{e}^{\theta_i S_i}] - 1\}}{\theta_i E[S_i]}\right)$$

provided the denominator is positive. To satisfy the latter requirement, the rate of input to server i must, in general, be smaller than the input allowed by the expected long-run waiting time formula in Eq. (3.1). Of course,

$$E\left[\mathrm{e}^{\theta_i S_i}\right] = \sum_{j=1}^{J} p_j a_{ji} E\left[e^{\theta_i S_{ji}}\right] \Big/ \sum_{j=1}^{J} p_j a_{ji}$$

can quickly grow large, or become formally infinite, depending on the assignment variables a_{ji}. Following this route, the objective function for problem P is defined to be

$$f(\underline{a}) = \sum_{j=1}^{J}\sum_{i=1}^{I} p_j b_j a_{ji} E[\mathrm{e}^{\theta_i W_i}] E[\mathrm{e}^{\theta_i S_{ji}}].$$

This objective function differs significantly from previously mentioned system performance measures, all of which are linear in the individual task or server performance measures. It is also possible to define a system measure as the maximum of the individual measures, e.g., $f(\underline{a}) = \max_j\{E[D_j]\}$. This type of system measures tends to yield an optimal assignment, which renders the individual measures, e.g., the expected delay for all task types, equal.

In conclusion, we propose a static model for randomly allocating or assigning incoming tasks to unequally capable servers. Because of the random assignment, standard results from queuing theory can be used to obtain closed form expressions for the total expected delay under two different scheduling strategies. One is to schedule the tasks for processing according to their arrival order and the other is to allow each server to nonpreemptively schedule his or her workload in an optimal manner. The problem of finding probability assignments that minimize the total expected

delay is posed as a mathematical program which is relatively simple and can be solved by standard optimization software, reinforced by heuristic methods to deal with the non-convexity of the total expected delay functions.

Acknowledgements

The authors are grateful to the referees for their helpful comments and suggestions. Professor Glazebrook would like to express appreciation for the support of the Engineering and Physical Sciences Research Council by means of grant GR/M09308.

References

Berman, O., Chiu, S.S., Larson, R.C., Odoni, A.R., Batta, R., 1990. Location of mobile units in a stochastic environment. In: Mirchandani, P.B., Francis, R.L. (Eds.), Discrete Location Theory. Wiley, New York, pp. 503–549.

Brooke, A., Kendrick, D., Meeraus, A., 1992. GAMS: A User's Guide, Release 2.25, The Scientific Press, South San Francisco, CA.

Eager, D.L., Lazowska, E.D., Zahorjan, J., 1986a. Adaptive load sharing in homogeneous distributed systems. IEEE Transactions on Software Engineering 12, 662–675.

Eager, D.L., Lazowska, E.D., Zahorjan, J., 1986b. A comparison of receiver-initiated and sender-initiated adaptive load sharing. Performance Evaluation 6, 53–68.

Fife, D.W., 1965. Scheduling with random arrivals and linear loss functions. Management Science 11 (3), 429–437.

Gaver, D.P., Morrison, J.A., Silveira, R., 1993. Service-adaptive multitype repairman problems. SIAM Journal on Applied Mathematics 53 (2), 459–470.

Gelenbe, E., Mitrani, I., 1980. Analysis and Synthesis of Computer Systems. Academic Press, London.

Green, L., 1985. A queuing system with general-use and limited-use servers. Operations Research 33, 168–182.

Kleinrock, L., 1976. Queueing Systems, Vol. II, Wiley, New York.

Mehrotra, V., 1997. Thank you for calling. How may I assist you? Department Seminar, Operations Research Department, Naval Postgraduate School, Monterey, CA, May 28.

Ross, K.W., Yao, D.D., 1991. Optimal load balancing and scheduling in a distributed computer system. Journal of the Association for Computing Machinery 38 (3), 676–690.

Shanthikumar, J.G., Yao, D.D., 1992. Multi-class queuing systems: polymatroidal structure and optimal scheduling control. Operations Research. 40 (2), S293–S299.

Stanford, D.A., Grassmann, W.K., 1993. The bilingual server model: a queueing model featuring fully and partially qualified servers. Information Systems and Operational Research 31 (4), 261–277.

Zhou, S., 1988. A trace-driven simulation study of dynamic load balancing. IEEE Transactions on Software Engineering 14, 1327–1341.